Supplementary Material

Overview of the Appendix

The main contents of this appendix are as follows:

- App. A: Presents overview of NCD, efficiency analysis of SAM, and theoretical results and proof regarding theorems.
- App. B: Describes detailed experimental settings.
- App. C: Effectiveness on S1-S4 Search Spaces.
- App. D: Presents strong generalizability on OoD-ViT-NAS-Ti search space.
- App. E: Presents visualization of architectures searched by our method.
- App. F: Presents more discussion and experimental results.
- App. G: Presents related work.

A. Overview of NCD, Efficiency Analysis of SAM, and Theoretical Analysis

A.1. Overview of NCD

To fulfill the goal of enhancing the training-free NAS system by defying negative correlation, NCD comprises a *stochastic activation masking* and a *non-linear rescaling*. The framework of the proposed NCD is provided in App. 1. First, we provide the definition of variables to illustrate our method for training-free NAS system in detail. To be specific, $\mathcal{A}$ denotes architecture search space, $\mathcal{N}$ represents number of architecture sampling, $\mathcal{T}$ denotes task, $\mathcal{D}$ denotes dataset, $\mathcal{B}$ denotes batch size, $F_{best}(\cdot)$ denotes searched best architecture by our NCD. Second, we select a batch $\mathcal{B}$ input images $X = \{s_i\}_{i=1}^{\mathcal{B}}$ from dataset $\mathcal{D}$ for task $\mathcal{T}$. Third, we randomly select an architecture $F_j(\cdot)$ from architecture search space $\mathcal{A}$, then calculate activation patterns c_i by obtaining $F_j(s_i)$. After that, we score the architecture $F_j(\cdot)$ using NCD. Finally, we can achieve the final searched architecture $F_{best}(\cdot)$ by selecting the highest $Score_{best}$ scored by NCD.

Algorithm 1 The Framework of NCD Method

```
procedure TRAINING-FREE NAS SYSTEM
    Input: Given task $\mathcal{T}$ and dataset $\mathcal{D}$, architecture search space $\mathcal{A}$, number of architecture sampling $\mathcal{N} \in \mathcal{A}$, batch size $\mathcal{B}$.
    Output: Best architecture $F_{best}(\cdot)$.
    Initialization: $Score_{best} \leftarrow 0$, $F_{best}(\cdot) \leftarrow 0$.
    for $j$ to $[1, \mathcal{N}]$ do
        $a_j \leftarrow$ Randomly select a batch $\mathcal{B}$ input images $X = \{s_i\}_{i=1}^{\mathcal{B}}$ from dataset $\mathcal{D}$ for task $\mathcal{T}$;
        $F_j(\cdot) \leftarrow$ Randomly select an neural architecture from search space $A$;
        $Score_j \leftarrow$ Calculate activation patterns $c_i$ by obtaining $F_j(s_i)$, then score the architecture $F_j(\cdot)$ using NCD;
        if $Score_{best} \geq Score_j$ then
            $Score_{best} \leftarrow Score_j$;
            $F_{best}(\cdot) \leftarrow F_{(}\cdot)$;
        end
    end
    Return $F_{best}(\cdot)$.
end procedure
```

A.2. Detailed Proof for Theorems

A.2.1. The Detailed Proof for Theorems 4.2

Theorem A.2. *Given a convolution output tensor* $\mathbf{X} \in \mathbb{R}^{B \times C \times H \times W}$*, the calculation of* $\mu_{(i,:,:,:)}$ *within the Layer Normalization(LN) of value* $x_{(i,j,k,p)}$ *can be formulated as:*

$$\mu_{(i,:,:,:)} = \frac{\sum_{a=1}^{H} \sum_{b=1}^{W} \sum_{c=1}^{C} \sum_{d=\lfloor -\frac{K}{2} \rfloor}^{K} \sum_{e=\lfloor -\frac{K}{2} \rfloor}^{K} \hat{x}_{(i,c,a+d,b+e)}}{H \cdot W}, \\ \hat{x}_{(i,c,a+d,b+e)} = \mathbf{E}[w] \cdot \sigma(x^{input}_{(i,c,j+c,p+d)}), \tag{1}$$

where $\mathbf{E}[w]$ *represents the expectation of the kernel weight* w.

Proof. Since the $\mu_{(i,:,:,:)}$ in LN is calculated within the feature map dimensions $C \times H \times W$, which can be formulated as follows:

$$\mu_{(i,:,:,:)} = \frac{\sum_{a=1}^{H} \sum_{b=1}^{W} \sum_{\hat{c}=1}^{C} x^{output}_{(i,\hat{c},a,b)}}{C \cdot H \cdot W}, \tag{2}$$

where $x^{output}_{(a,j,k,p)}$ is the value of the output feature map resulting from the convolution operation. Similar to the proof of Theorem 4.1, $x^{output}_{(a,j,k,p)}$ can be defined as:

$$x^{output}_{(i,\hat{c},a,b)} = \sum_{c=1}^{C} \sum_{d=\lfloor -\frac{K}{2} \rfloor}^{K} \sum_{e=\lfloor -\frac{K}{2} \rfloor}^{K} \hat{x}_{(i,c,a+d,b+e)},$$
$$\hat{x}_{(i,c,a+d,b+e)} = \omega_{(c,\hat{d}-\lfloor -\frac{K}{2} \rfloor, e-\lfloor -\frac{K}{2} \rfloor)} \cdot \sigma(x^{input}_{(i,c,a+d,b+e)}), \tag{3}$$

where we assume the input and output feature map have same number of channel. Note that the $x^{output}_{(i,\hat{c},a,b)}$ of every channel are calculated by applying different filter kernel to the same perceptive field of the feature map, and accordingly, we have:

$$x^{output}_{(i,\sum,a,b)} = \sum_{\hat{c}=1}^{C} \sum_{c=1}^{C} \sum_{d=\lfloor -\frac{K}{2} \rfloor}^{K} \sum_{e=\lfloor -\frac{K}{2} \rfloor}^{K} \hat{x}^{\hat{c}}_{(i,c,a+d,b+e)},$$
$$\hat{x}^{\hat{c}}_{(i,c,a+d,b+e)} = \omega^{\hat{c}}_{(c,\hat{d}-\lfloor -\frac{K}{2} \rfloor, e-\lfloor -\frac{K}{2} \rfloor)} \cdot \sigma(x^{input}_{(i,c,a+d,b+e)}). \tag{4}$$

By reformulating the $x^{output}_{(i,\sum,a,b)}$, we can rewrite Eq. 4 as follows:

$$x^{output}_{(i,\sum,a,b)} = \sum_{c=1}^{C} \sum_{d=\lfloor -\frac{K}{2} \rfloor}^{K} \sum_{e=\lfloor -\frac{K}{2} \rfloor}^{K} \hat{x}^{\sum}_{(i,c,a+d,b+e)},$$
$$\hat{x}^{\sum}_{(i,c,a+d,b+e)} = \sum_{\hat{c}=1}^{C} \omega^{\hat{c}}_{(c,\hat{d}-\lfloor -\frac{K}{2} \rfloor, e-\lfloor -\frac{K}{2} \rfloor)} \cdot \sigma(x^{input}_{(i,c,a+d,b+e)}). \tag{5}$$

We notice that:

$$\sum_{\hat{c}=1}^{C} \omega^{\hat{c}}_{(c,\hat{d}-\lfloor -\frac{K}{2} \rfloor, e-\lfloor -\frac{K}{2} \rfloor)} = C \cdot \mathbf{E}[w]. \tag{6}$$

Consequently, by substituting Eq.5 and Eq.6 into Eq.2, we can write:

$$\mu_{(i,:,:,:)} = \frac{\sum_{a=1}^{H} \sum_{b=1}^{W} \sum_{c=1}^{C} \sum_{d=\lfloor -\frac{K}{2} \rfloor}^{K} \sum_{e=\lfloor -\frac{K}{2} \rfloor}^{K} \hat{x}_{(i,c,a+d,b+e)}}{H \cdot W}.$$
$$\hat{x}_{(i,c,a+d,b+e)} = \mathbf{E}[w] \cdot \sigma(x^{input}_{(i,c,j+c,p+d)}). \tag{7}$$

□

Therefore, Theorem 4.2 is proved.

A.3. Efficiency Analysis of SAM

Since SAM randomly masks each activation value by 0 with probability α, it can speed up the network forward process of AZP computation. As analysed in [19], when a filter is applied to the local receptive field of the input feature map to produce an output value, the computational cost $\mathcal{R}$ can be formulated as:

$$\mathcal{R} = d_k \cdot d_k \cdot c_{in}, \tag{8}$$

where d_k is the size of the filter kernel, c_{in} is the channel of the input feature map. By utilizing SAM, approximately α proportion of the input values within the local receptive field are set to 0 and cut from the computation process. Therefore, the computational cost $\mathcal{R}_{SAM}$ can be modified as follows:

$$\mathcal{R}_{SAM} = (1 - \alpha) \cdot d_k \cdot d_k \cdot c_{in}. \tag{9}$$

From the above analysis, we can find that SAM only requires $1 - \alpha$ times of the original computational cost during convolution and brings high calculation efficiency to AZPs.

B. Detailed Experimental Settings

B.1. Experimental Settings

Search spaces. We use *ten search spaces* to validate the advantages of NCD. To the best of our knowledge, we are the first work to comprehensively evaluate ten existing search spaces in the NAS field. To be specific, DARTS search space [30] contains 10^{18} architectures, and consists of 7 representative operations: (1) zero, (2) skip connection, (3) 3 × 3 dilated separable convolutions, (4) 3 x 3 separable convolutions, (5) 5 × 5 separable convolutions, (6) 5 × 5 dilated separable convolutions, and (7) 3 x 3 average pooling layer. Due to NAS-Bench-310 utilizing the same architectures as DARTS search space, we do not provide the experiments on NAS-Bench-310.

NAS-Bench-101 search space [57] contains 423624 architectures, and consists of 3 representative operations: (1) 1 x 1 convolution, (2) 3 x 3 convolution, and (3) 3 x 3 max pooling layer.

NAS-Bench-201 search space [13] contains 15625 architectures, and consists of 5 representative operations: (1) zero, (2) skip connection, (3) 1 x 1 convolution, (4) 3 x 3 convolution, and (5) 3 x 3 average pooling layer.

TransNAS-Bench-101-Mirco/Macro [14] consists of a micro (cell-based, 4096 architectures) search space and a macro (stack-based, 3256 architectures) search space, which use 7 representative operations: (1) Zeroize, (2) skip connection, (3) 1 × 1 convolutions, (4) 3 x 3 convolutions.

S1 search space uses a distinct set of only two operators on each edge, which is generated by an offline process that iteratively removes the least important operations from the

DARTS search space. S2 search space consists of 2 operations: (1) 3 x 3 separable convolutions, (2) skip connection. S3 search space consists of 3 operations: (1) 3 x 3 separable convolutions, (2) skip connection, (3) zero. S4 search space consists of 2 operations: (1) 3 x 3 separable convolutions, (2) Noise. The Noise operation replaces each value in the input feature map with noise $\epsilon \sim \mathcal{N}(0, 1)$.

The MobileNet-like search space is modified by the architecture of MobileNetV2 [42], which consists of MobileNet blocks. The main search component is the expansion ratio at the depth-wise level, i.e., $\{1, 2, 4, 6\}$.

Evaluation Tasks. *Four real-world tasks* are used to validate the effectiveness of NCD. The details are as follows:

- **Image recognition tasks:** We evaluate NCD in CIFAR-10/100[26], ImageNet16-120[8] and ImageNet-1k [11] datasets.
- **Autoencoding task[25]:** We evaluate NCD on a pixel-level prediction task in Taskonomy dataset [58], which reconstructs the image by obtaining the latent representation of the origin image with searched architecture.
- **Scene classification task:** We evaluate NCD in MIT Places dataset [62].
- **Self-supervised jigsaw puzzle task[25]:** We evaluate NCD in Taskonomy dataset [58].

Peer Competitors. To fairly compare the performance with previous methods, in this paper, we only consider published works for performance comparison, the papers from arXiv are not compared according to the rule of ICCV 2025.

For NAS-Bench-201 search space with CIFAR-10/100, and ImageNet16-120 datasets, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) Params; (2) FLOPs; (3) Snip [1]; (4) Grasp [1, 48]; (5) Synflow [1, 47]; (6) ZenNAS [29]; (7) ZiCo [28]; (8) AZ-NAS [27]; (9) SWAP [39]; (10) NWOT [35].

For NAS-Bench-101 search space with CIFAR-10 dataset, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) Params; (2) FLOPs; (3) Snip [1]; (4) Grasp [1, 48]; (5) Synflow [1, 47]; (6) ZenNAS [29]; (7) ZiCo [28]; (8) AZ-NAS [27]; (9) SWAP [39]; (10) NWOT [35].

Notably, ParZC [12] is the hybrid method, Auto-Prox [51] is designed for Transformer, therefore, we do not compare ParZC [12] and Auto-Prox [51] with our method on NAS-Bench-201&101 search spaces.

For DARTS search space in CIFAR-10/100, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) ResNet18 [17]; (2) DenseNet-BC [21]; (3) AmoebaNet-A [64]; (4) NASNet-A [41]; (5) NASNet-A [41]; (6) ENAS [40]; (7) SNAS [54]; (8) DARTS(2nd) [30]; (9) DARTS+PT [49]; (10) DARTS(1nd) [30]; (11) DARTS- [9]; (12) FairDARTS-D [10]; (13) β-DARTS [56]; (14) P-DARTS [7]; (15) PC-DARTS [55]; (16) NASWOT [35]; (17) NASI-ADA [44]; (18) TENAS [5]; (19) Random; (20) DARTS- [9]; (21) DARTS+PT [49]; (22) Λ-DARTS [37]; (23) FP-DARTS [50]; (24) DARTS-AER^b [24]; (25) IS-DARTS [16].

For TransNAS-Bench-101-Micro/Macro search space, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) Grad_norm [1]; (2) SNIP [1]; (3) Grasp [1, 48]; (4) Fisher [1, 48]; (5) Synflow [1, 47]; (6) Zen-score [29]; (7) GradSign [61]; (8) Params; (9) FLOPs; (10) ZiCo [28]; (11) SWAP [39]; (12) NWOT [35].

For S1-S4 search spaces, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) DARTS [30];(2) R-DARTS [59]; (3)PC-DARTS [55]; (4)DARTS- [9] ; (5) SDARTS [6]; (6) DARTS+PT [49].

For MobileNet-like search space, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) PloxylessNAS [3]; (2) FBNet-C [53]; (3) FairNAS-A [10]; (4) FairDARTS-D [10]; (5) RLNAS [60]; (6) GM+ProxylessNAS [20]; (7) OLES [23].

For AutoFormer search space, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) ViT-Ti [**?**]; (2) NWOT [35]; (3) AutoFormer-T [4]; (4) ViTAS-C [45]; (5) TF-TAS-T [63]; (6) Auto-Prox [51]; (7) AZ-NAS [27]; (8) ParZC [12]. Notably, Auto-Prox [51] utilizes evolution to search target proxy for CNNs, ParZC [12] is a combination of evolution, random search, reinforcement, gradient, and training-free for CNNs.

For OoD-ViT-NAS-Ti [18] search space with ImageNet1k, ImageNet-A, ImageNet-R, ImageNet-D/Texture, and ImageNet-D/Material datasets, we compare our method with a wide scope of the state-of-the-art baselines, as follows: (1) Snip [1]; (2) Grasp [1, 48]; (3) MeCo; (4) CroZe; (5) DSS; (6) Auto-Prox [51]; (7) NWOT [35].

Parameter Settings. For searching on the DARTS, TransNAS-Bench-101-Micro/Macro, NAS-Bench-201 and NAS-Bench-101, S1-S4, MobileNet-like search spaces, we use random search as our experimental strategy, and the experimental settings are the same as AZ-NAS [27], which sample 3000 candidate architectures utilized for evaluation. The final architecture is selected with the highest value scored by our method.

For the performance in terms of accuracy on TransNAS-Bench-101-Micro/Macro, NAS-Bench-201, and NAS-Bench-101 search spaces, we directly obtain by retrieving the final architecture from benchmarks. Due to DARTS and MobileNet-like search space not providing such benchmarks, we need to retrain the searched architecture in specific datasets. The details are as follows:

(1) For training searched architecture on the DARTS search space in CIFAR-10/100 and ImageNet datasets, we use the same experimental settings as used with [30].

(2) For training searched architecture on the S1-S4 search spaces in CIFAR-10 dataset, we use the same ex-

Table 1. The test error (%) on S1-S4 search space in CIFAR-10 dataset.

Search space	DARTS [30]	R-DARTS(DP) [59]	R-DARTS(L2)	PC-DARTS [55]	DARTS+PT [49]	DARTS+ES [30]	DARTS+ADV [30]	SDARTS+ES [6]	SDARTS+ADV [6]	DARTS- [9]	NCD-NWOT ($\alpha = 0.5$)
S1	3.84	3.11	2.78	3.11	3.50	3.01	3.10	2.78	2.73	2.68	**2.52**↓(0.16)
S2	4.85	3.48	3.31	3.02	2.79	3.26	3.35	2.75	2.65	2.63	**2.62**↓(0.01)
S3	3.34	2.93	2.51	2.51	2.49	2.74	2.61	2.53	2.49	2.42	2.59
S4	7.20	3.58	3.56	3.02	2.64	3.71	4.84	2.93	2.87	2.86	**2.83**↓(0.03)

Table 2. Correlation on OoD-ViT-NAS-Ti search space.

Method	ImageNet1k	ImageNet-A	ImageNet-R	ImageNet-D/Texture	ImageNet-D/Material
SNIP	0.38	0.51	0.55	-0.06	0.11
Grasp	-0.03	-0.06	-0.07	-0.01	0.03
MeCo	0.48	0.40	0.33	0.09	0.08
CroZe	0.40	0.54	0.60	0.01	0.12
DSS	0.62	0.82	0.81	0.02	0.17
AutoProx	0.67	0.82	0.78	0.05	0.15
NWOT	0.75	0.76	0.74	0.11	0.12
NWOT+SAM($\alpha = \mathbf{0.9}$)	0.77↑(0.02)	0.78	0.77	0.12↑(0.01)	0.13

perimental settings as used with PC-DARTS [55].

(3) For training searched architecture on the MobileNet-like search space in ImageNet1k dataset, we use the same experimental settings as used with RLNAS [60].

(4) For training searched architecture on the AutoFormer search space in the ImageNet1k dataset, we use the same experimental settings as used with TF-TAS-T [63].

Evaluation metrics. We use Top-1/5 (%) accuracy or Test Err. (%), Search Cost (GPU-days/hours), Runtime (ms), Mean Average Precision (mAP), Mean Intersection over Union (mIoU), number of parameters (M), FLOPs (M), the correlation coefficients in terms of Spearman's $\rho \in [-1, 1]$, Structural Similarity (SSIM), and COCO-style Average Precision (AP), as our evaluation metrics.

Codes. We implement our paper using Python 3.8 and PyTorch 2.2.0 with CUDA 12.1.

C. Effectiveness on S1-S4 Search Spaces

To validate the effectiveness of our method on S1-S4 search spaces, we conduct the experiment on the standard benchmarks from R-DARTS (S1-S4). R-DARTS includes four search spaces, i.e., S1, S2, S3, and S4. In this experiment, the optimal architecture is obtained by selecting the highest value searched by our method, after that, we train the searched architecture in the CIFAR-10 datasets by utilizing the same hyper-parameter settings in R-DARTS. As depicted in Table 1, we can observe that our method achieves optimal accuracy on S1, S2, and S4, which validates the effectiveness of our method on S1-S4 search spaces.

D. Strong Generalizability on OoD-ViT-NAS-Ti Search Space

To further scrutinize the generalizability of our method for vision transformer under Out-of-Distribution (OoD) shifts, we perform the empirical validation on OoD-ViT-NAS-Ti [18] search space in ImageNet1k, ImageNet-A, ImageNet-R, ImageNet-D/Texture, and ImageNet-D/Material datasets. To be specific, we report the results over 5 independent runs. Notably, vision transformer uses LN to accelerate model learning speed and improve training stability, therefore, we only compare the Spearman's ρ correlation between SAM with AZP (i.e., NWOT) and other zero-cost proxies. As depicted in Table 2, we can clearly observe that our method obtains superior performance in terms of Spearman's ρ correlation under OoD shifts. In particular, "NWOT+SAM" achieves a maximum 0.02% higher Spearman's ρ in ImageNet1k and a maximum 0.01% higher Spearman's ρ in ImageNet-D/Texture than state-of-the-art methods (i.e., DSS, AutoProx). Those results demonstrate that our method possesses strong generalizability under OoD shifts.

E. Visualization of Searched Architectures

For better understanding, the visualization of architectures discovered by our NCD is also provided in Fig. 1 and Fig. 2. For DARTS search spaces, the optimal architectures discovered by NCD-NWOT in the CIFAR-10 dataset are shown in Fig. 1a and Fig. 1b. In addition, Fig. 2a and Fig. 2b show the optimal architectures found by NCD-SWAP in the CIFAR-10 dataset. As shown, we can clearly find that optimal architectures in the DARTS search space commonly possess more inception structures and skip connections, which can be sufficiently explored by our NCD-NWOT and NCD-SWAP.

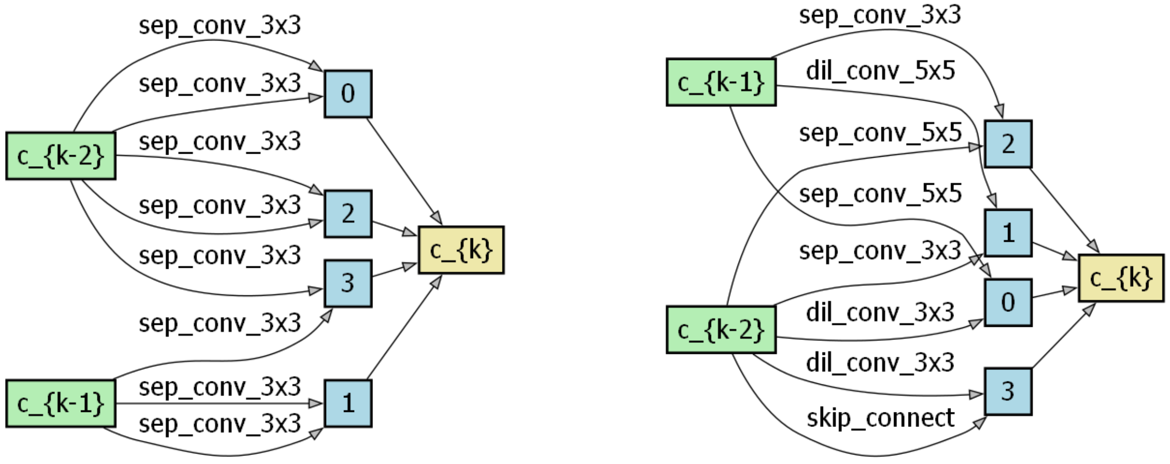
F. More Discussion and Experimental Validations

F.1. Validation of Negative Correlation on More Search Spaces

Changes of correlations (negative correlation) of AZP are a general issue for NAS search spaces and datasets. Due to the main page limit, we only provide the visualization on the representative NAS-Bench-201 search space in the CIFAR-10 dataset. To enhance our statement, we provide additional results on **TransNAS-Bench-101** (as shown in

Table 3. Accuracy comparison between NIR and AZP methods.

Method	NB101-CIFAR10	Tras101/Micro-Autoencoding
NWOT	93.16±0.36	46.40±0.70
NWOT+NIR	93.32±0.27↑(0.16)	51.78±3.68↑(5.18)
SWAP	90.51±2.08	43.09±1.69
SWAP+NIR	92.77±1.24↑(2.26)	53.29±2.32↑(10.20)



(a) NCD-NWOT Normal Cell

(b) NCD-NWOT Reduce Cell

Figure 1. (a) and (b) are the best cells found by NCD-NWOT on DARTS search spaces in CIFAR-10 dataset with model size 4.4M.

Fig. 3), validating our statement.

F.2. Further Discussion of NIR in Training-free NAS

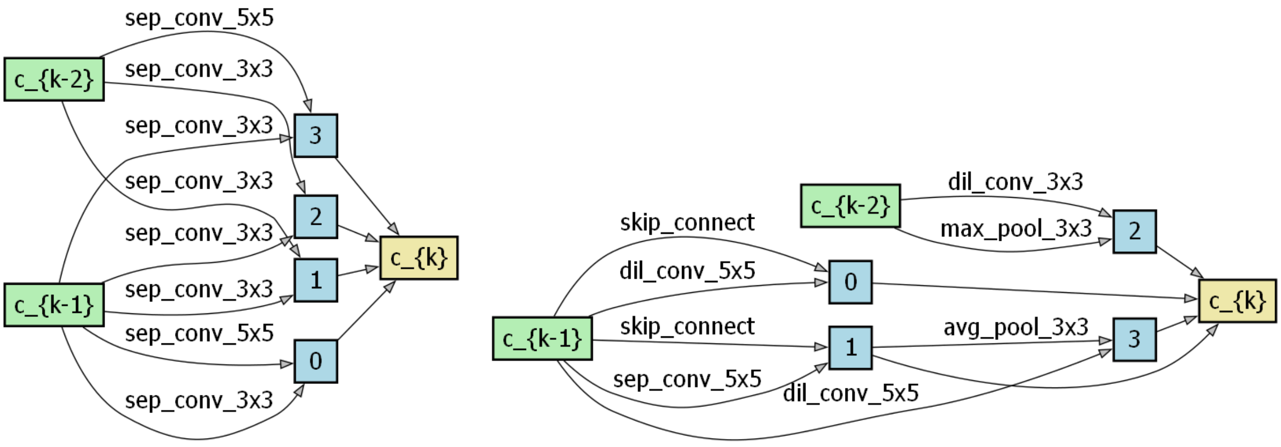
To further scrutinize the effectiveness of the proposed NIR in training-free NAS, we conduct in-depth analysis from empirical validation on NB201 with NB101 with CIFAR10, and Tras101 with Micro-Autoencoding. To be specific, we report the results over 5 independent runs. First, we only observe the superiority of LN for stability in AZP-based NAS during proxy evaluation. Second, additional experiments (as shown in Table 3) show the accuracy of architectures found with NIR better than BN in mainstream tasks. In the future, we will further study LN *vs.* BN in beyond proxy evaluation to better understand their behaviors.

F.3. Discussion about different activations

Our NCD is applicable to various activations, since its core idea is not ReLU-specific. As shown in Fig. 4, when the non-linearity level of GELU and LeakyReLU reaches the turning point, AZPs still show negative correlation. First, the increase in the non-linearity level of GELU and LeakyReLU is also brought by the sum of activation values, and the decrease in the scores of architectures with Conv3x3 to Conv5x5 can support this conclusion, as each output value of Conv5x5 is summed with more activation values than Conv3x3. Consequently, our method for negative correlation still works, since NCD's core idea is to reduce the quantity of activation values included in the sum so that the non-linearity level can return to before the turning point. Additionally, experiments on the Autoformer also show the effectiveness of NCD for the GELU activation pattern. Since modern architectures employ non-saturated activation functions with forms similar to ReLU, such as GELU and LeakyReLU mentioned, we believe the problem presence and analysis can be broadly applicable.

F.4. Further discussion about other normalizations

In this paper, NIR aims to reduce the number of activation values involved in the sum of normalization. In BN and IN, the mean μ can be seen as a weighted sum of activation values. For LN, each activation value is weighted by the expectation of weights participating in the calculation of input values, which is zero since the architecture is initialized. In GN, activation values are weighted by the expectation of weights participating in the calculation of input values within the corresponding group. As the number of groups decreases, more weights are included, and the expectation approaches zero. When there's one group, GN becomes LN. Thus, NIR is necessary in search spaces with BN, GN, and IN. The analyses of GN and IN follow Theorems 4.1 & 4.2, with detailed proofs to be added in the Appendix. Additional results (As shown in Table 4) validate our statement.



(a) NCD-SWAP Normal Cell (b) NCD-SWAP Reduce Cell

Figure 2. (a) and (b) are the best cells found by NCD-SWAP on DARTS search spaces in CIFAR-10 dataset with model size 4.3M.

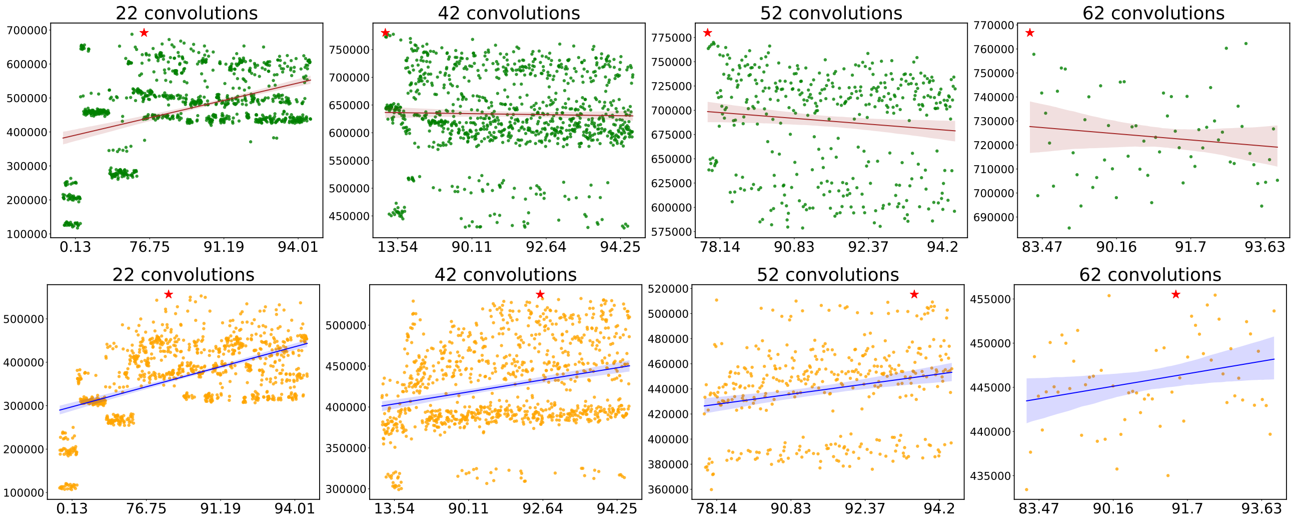


Figure 3. The scores of SWAP(top) and NCD-SWAP(bottom) on the Trans101-Micro-Jigsaw.

F.5. Scalability of NCD

App. A.2 shows SAM can proportionally reduce computational cost during the network forward process when AZP scoring, whose acceleration effect is similar to network pruning. This efficiency advantage enables AZPs to be applied to larger search spaces. Our analysis indicates that deeper or wider architectures may require a higher mask rate α, as later layers generate more nonlinear features that are more likely to exceed the turning point when α is too small. Table 5 shows that NCD can generalize to larger model sizes.

G. Related Work

In this section, we introduce the design and search approaches of network architecture, zero-cost proxies, and NAS methods that are most relevant to this work.

G.1. Design & Search Neural Architecture

NAS can automatically design high-performance neural architecture for real-world tasks (i.e., image recognition, scene classification) by adopting reinforcement learning [3, 41, 46], and evolutionary algorithm [34, 40, 64]. However, those methods suffer from huge computational budgets due to training networks iteratively. To reduce the search costs, one-shot approaches [2, 15, 32] are proposed, which use a weight sharing supernet to reduce the training time of each potential subnet. Moreover, gradient descent based methods [30] are proposed, where the network weights and architecture parameters are optimized alternately in a differentiable way, including DARTS [30], PC-DARTS [55], IS-DARTS [16], etc. For example, IS-DARTS [16] reduces the search cost to 0.42 GPU Days on DARTS search space in CIFAR-10 dataset. Although differentiable

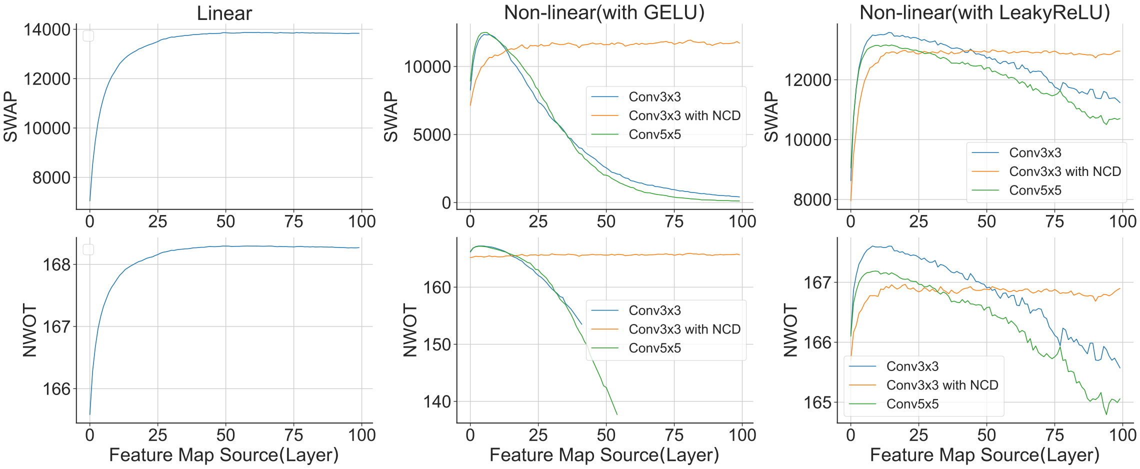


Figure 4. Visualization of the negative correlation problem with different activation patterns(GELU, LeakyReLU), where the score variation curves of NWOT with GELU are incomplete since it encounters the numerical overflow issue.

Table 4. A comparison between normalizations and our NIR (LN) of AZPs.

Method	NB201-ImageNet	NB201-CIFAR10	Tras101/Micro-Autoencoding
NWOT+BN	42.31±3.43	91.95±1.29	46.40±0.70
NWOT+GN(group=16)	42.98±3.85	92.24±0.01	47.59±0.38
NWOT+GN(group=8)	43.13±1.48	93.37±0.29	48.68±0.42
NWOT+IN	36.83±1.63	91.32±1.59	37.02±0.31
NWOT+Dropout	40.55± 4.15	91.41±0.18	44.01±0.62
NWOT+NIR	43.75±1.91	93.48±0.50	51.78±1.68
SWAP+BN	35.40±3.96	90.48±0.94	43.09±1.69
SWAP+GN(group=16)	37.42±2.01	91.35±0.39	46.14±0.17
SWAP+GN(group=8)	41.80±1.96	92.31±0.48	46.56±0.23
SWAP+IN	21.84±1.62	86.35±0.33	35.01±0.51
SWAP+Dropout	34.61±3.79	92.35±0.33	45.10±0.73
SWAP+NIR	44.07±1.36	93.01±0.54	53.29±1.32

methods show remarkable performance in terms of accuracy and search speed, they still need to train a supernet for each dataset. In addition, differentiable methods suffer from the performance collapse issue, which will significantly affect the performance. This highlights the need to liberate NAS from aforementioned bottlenecks.

G.2. Efficient and Training-free NAS

To facilitate the development of the NAS community, several NAS benchmarks (i.e., NAS-Bench-101 [57], NAS-Bench-201 [13]) are proposed for computer vision tasks (i.e., image classification[43]), which include the ground truth of each candidate architecture in the target vision task. Moreover, TransNAS-Bench-101-Mirco/Macro [14] is built, which provides the ground truth of each candidate architecture across seven tasks, including autoencoding task, scene classification task, self-supervised jigsaw puzzle task, etc. Grounded in these benchmarks, predictor-based methods [31, 33, 52] are proposed, which can directly predict the accuracy of candidate architectures by building a GCN-based predictor.

Recent Training-free NAS methods significantly reduce computational costs by measuring the expressibility of neural architectures without training. Essentially, training-free NAS methods utilize some zero-cost proxies, predicting the accuracy ranking of neural architectures without training. These methods can be categorized into four groups based on how to represent the architecture. (1) **Gradient-based methods** utilize the indicators of pruning to estimate the accuracy ranking of neural architectures, including Snip [1], Grasp [1, 48], Synflow [1, 47]. For example, AZ-NAS [27] explore the ensemble zero-cost proxies in the views of expressivity, progressivity, trainability, and complexity, which provide an ensemble perspective to analyze how to design zero-cost proxy. However, those methods still require high search costs due to relying on additional backward or forward passes with inputs. (2) **Theory-based methods** [5, 36] use neural tangent kernel (NTK) [22] as indicator to assess expressivity of neural architectures. However, calculating NTK is difficult and needs large computational resources, making it hard to deploy those methods on large networks and datasets. (3) **Statistical-based methods** utilize statistical information (i.e., number of parameters or floating-point operations) of the architecture to measure the accuracy ranking. Although those methods seem very simple without backward or forward passes, which obtain better performance than well-designed proxies (i.e., Grasp, ZiCo [28]). (4) **Activation-based zero-cost proxies** (AZP)

Table 5. Scalability of NCD to larger model sizes on AutoFormer.

Base search space	Swin-B	NWOT	AutoFormer-B	GLiT-B	TF-TAS-B	Auto-Prox	AZ-NAS	ParZC	**NWOT+SAM**($\alpha = 0.9$)
Param (M)	88.0	67.2	54.0	54.0	54.0	-	54.1	-	53.4
Search Cost (GPU Days)	-	0.03	24	-	0.5	-	0.07	-	0.02
Top-1 (%)	83.5	83.6	82.4	82.3	82.2	-	82.3	-	84.5

[35, 38] are proposed by analyzing the activation patterns of candidate architectures, which achieve superior performance in terms of accuracy and search costs without additional gradient descent. However, we identify a distinctive negative correlation issue of AZP, which significantly impacts the accuracy of AZP-based NAS. In this paper, we overcome this issue and outperform SOTA methods on 12 search spaces with 4 tasks by proposing a simple yet effective negative correlations-defied AZP-based NAS. Experimental results demonstrate that our approach successfully addresses the negative correlation issue in AZP and significantly improves performance across various benchmarks.

References

[1] Mohamed S. Abdelfattah, Abhinav Mehrotra, Lukasz Dudziak, and Nicholas Donald Lane. Zero-cost proxies for lightweight NAS. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021. 3, 7

[2] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In International Conference on Machine Learning, pages 550–559. PMLR, 2018. 6

[3] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. ArXiv, abs/1812.00332, 2018. 3, 6

[4] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. In International Conference on Computer Vision, 2021. 3

[5] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. ArXiv, abs/2102.11535, 2021. 3, 7

[6] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In International Conference on Machine Learning, pages 1554–1565. PMLR, 2020. 3, 4

[7] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1294–1303, 2019. 3

[8] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. arXiv preprint arXiv:1707.08819, 2017. 3

[9] Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. Darts-: robustly stepping out of performance collapse without indicators. ArXiv, abs/2009.01027, 2020. 3, 4

[10] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair darts: Eliminating unfair advantages in differentiable architecture search. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV, pages 465–480. Springer, 2020. 3

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255, 2009. 3

[12] Peijie Dong, Lujun Li, Xinglin Pan, Zimian Wei, Xiang Liu, Qiang Wang, and Xiaowen Chu. Parzc: Parametric zero-cost proxies for efficient nas. In Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI-2025), 2025. 3

[13] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, 2020. 2, 7

[14] Yawen Duan, Xin Chen, Hang Xu, Zewei Chen, Xiaodan Liang, Tong Zhang, and Zhenguo Li. Transnas-bench-101: Improving transferability and generalizability of cross-task neural architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5251–5260, 2021. 2, 7

[15] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In Computer vision–ECCV 2020: 16th European conference, glasgow, UK, August 23–28, 2020, proceedings, part XVI 16, pages 544–560. Springer, 2020. 6

[16] Hongyi He, Longjun Liu, Haonan Zhang, and Nanning Zheng. Is-darts: Stabilizing darts through precise measurement on candidate importance. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 12367–12375, 2024. 3, 6

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016. 3

[18] Sy-Tuyen Ho, Tuan Van Vo, Somayeh Ebrahimkhani, and Ngai man Cheung. Vision transformer neural architecture search for out-of-distribution generalization: Benchmark and insights. In The Thirty-eighth Annual Conference on Neural Information Processing Systems, 2024. 3, 4

[19] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017. 2

[20] Shoukang Hu, Ruochen Wang, Lanqing Hong, Zhenguo Li, Cho-Jui Hsieh, and Jiashi Feng. Generalizing few-shot nas with gradient matching. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2022. 3

[21] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4700–4708, 2017. 3

[22] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. Advances in Neural Information Processing Systems, 31, 2018. 7

[23] Shen Jiang, Zipeng Ji, Guanghui Zhu, Chunfeng Yuan, and Yihua Huang. Operation-level early stopping for robustifying differentiable nas. Advances in Neural Information Processing Systems, 36, 2023. 3

[24] Kun Jing, Luoyu Chen, and Jungang Xu. An architecture entropy regularizer for differentiable neural architecture search. Neural Networks, 158:111–120, 2023. 3

[25] Arjun Krishnakumar, Colin White, Arber Zela, Renbo Tu, Mahmoud Safari, and Frank Hutter. Nas-bench-suite-zero: Accelerating research on zero cost proxies. Advances in Neural Information Processing Systems, 35:28037–28051, 2022. 3

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Toronto, ON, Canada, 2009. 3

[27] Junghyup Lee and Bumsub Ham. Az-nas: Assembling zero-cost proxies for network architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5893–5903, 2024. 3, 7

[28] Guihong Li, Yuedong Yang, Kartikeya Bhardwaj, and Radu Marculescu. Zico: Zero-shot nas via inverse coefficient of variation on gradients. arXiv preprint arXiv:2301.11300, 2023. 3, 7

[29] Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-nas: A zero-shot NAS for high-performance image recognition. In 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021, pages 337–346. IEEE, 2021. 3

[30] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. ArXiv, abs/1806.09055, 2018. 2, 3, 4, 6

[31] Yuqiao Liu, Yehui Tang, Zeqiong Lv, Yunhe Wang, and Yanan Sun. Bridge the gap between architecture spaces via a cross-domain predictor. Advances in Neural Information Processing Systems, 35:13355–13366, 2022. 7

[32] Zhichao Lu, Gautam Sreekumar, Erik Goodman, Wolfgang Banzhaf, Kalyanmoy Deb, and Vishnu Naresh Boddeti. Neural architecture transfer. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(9):2971–2989, 2021. 6

[33] L. Ma, H. Kang, G. Yu, Q. Li, and Q. He. Single-domain generalized predictor for neural architecture search system. IEEE Transactions on Computers, 73(05):1400–1413, 2024. 7

[34] Lianbo Ma, Nan Li, Peican Zhu, Keke Tang, Asad Khan, Feng Wang, and Guo Yu. A novel fuzzy neural network architecture search framework for defect recognition with uncertainties. IEEE Transactions on Fuzzy Systems, 2024. 6

[35] Joe Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. Neural architecture search without training. In International Conference on Machine Learning, pages 7588–7598. PMLR, 2021. 3, 8

[36] Jisoo Mok, Byunggook Na, Ji-Hoon Kim, Dongyoon Han, and Sungroh Yoon. Demystifying the neural tangent kernel from a practical perspective: Can it be trusted for neural architecture search without training? In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11861–11870, 2022. 7

[37] Sajad Movahedi, Melika Adabinejad, Ayyoob Imani, Arezou Keshavarz, Mostafa Dehghani, Azadeh Shakery, and Babak Nadjar Araabi. λ-darts: Mitigating performance collapse by harmonizing operation selection among cells. In The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023, 2023. 3

[38] Yameng Peng, Andy Song, Haytham M Fayek, Vic Ciesielski, and Xiaojun Chang. Swap-nas: Sample-wise activation patterns for ultra-fast nas. arXiv preprint arXiv:2403.04161, 2024. 8

[39] Yameng Peng, Andy Song, Haytham M. Fayek, Vic Ciesielski, and Xiaojun Chang. SWAP-NAS: Sample-wise activation patterns for ultra-fast NAS. In The Twelfth International Conference on Learning Representations, 2024. 3

[40] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In International Conference on Machine Learning, pages 4095–4104. PMLR, 2018. 3, 6

[41] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In AAAI Conference on Artificial Intelligence, page 4780–4789, 2019. 3, 6

[42] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4510–4520, 2018. 3

[43] Fei Shen, Xin Jiang, Xin He, Hu Ye, Cong Wang, Xiaoyu Du, Zechao Li, and Jinhui Tang. Imagdressing-v1: Customizable virtual dressing. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 6795–6804, 2025. 7

[44] Yao Shu, Shaofeng Cai, Zhongxiang Dai, Beng Chin Ooi, and Bryan Kian Hsiang Low. NASI: Label- and data-agnostic neural architecture search at initialization. In International Conference on Learning Representations, 2022. 3

[45] Xiu Su, Shan You, Jiyang Xie, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Vision transformer architecture search. In Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXI, page 139–157, 2022. 3

[46] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In International Conference on Machine Learning, pages 6105–6114. PMLR, 2019. 6

[47] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. Advances in Neural Information Processing Systems, 33:6377–6389, 2020. 3, 7

[48] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. arXiv preprint arXiv:2002.07376, 2020. 3, 7

[49] Ruochen Wang, Minhao Cheng, Xiangning Chen, Xiaocheng Tang, and Cho-Jui Hsieh. Rethinking architecture selection in differentiable nas. ArXiv, abs/2108.04392, 2021. 3, 4

[50] Wenna Wang, Xiuwei Zhang, Hengfei Cui, Hanlin Yin, and Yannnig Zhang. Fp-darts: Fast parallel differentiable neural architecture search for image classification. Pattern Recognition, 136:109193, 2023. 3

[51] Zimian Wei, Peijie Dong, Zheng Hui, Anggeng Li, Lujun Li, Menglong Lu, Hengyue Pan, and Dongsheng Li. Auto-prox: Training-free vision transformer architecture search via automatic proxy discovery. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 15814–15822, 2024. 3

[52] Wei Wen, Hanxiao Liu, Yiran Chen, Hai Li, Gabriel Bender, and Pieter-Jan Kindermans. Neural predictor for neural architecture search. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX, pages 660–676. Springer, 2020. 7

[53] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10734–10742, 2019. 3

[54] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: Stochastic neural architecture search. ArXiv, abs/1812.09926, 2020. 3

[55] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient architecture search. ArXiv, abs/1907.05737, 2019. 3, 4, 6

[56] Peng Ye, Baopu Li, Yikang Li, Tao Chen, Jiayuan Fan, and Wanli Ouyang. $\backslash beta$-darts: Beta-decay regularization for differentiable architecture search. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 10864–10873. IEEE, 2022. 3

[57] Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In International Conference on Machine Learning, pages 7105–7114. PMLR, 2019. 2, 7

[58] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3712–3722, 2018. 3

[59] Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. ArXiv, abs/1909.09656, 2019. 3, 4

[60] Xuanyang Zhang, Pengfei Hou, Xiangyu Zhang, and Jian Sun. Neural architecture search with random labels. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10907–10916, 2021. 3, 4

[61] Zhihao Zhang and Zhihao Jia. Gradsign: Model performance inference with theoretical insights. arXiv preprint arXiv:2110.08616, 2021. 3

[62] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(6):1452–1464, 2017. 3

[63] Qinqin Zhou, Kekai Sheng, Xiawu Zheng, Ke Li, Xing Sun, Yonghong Tian, Jie Chen, and Rongrong Ji. Training-free transformer architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10894–10903, 2022. 3, 4

[64] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8697–8710, 2018. 3, 6